AN EFFICIENT IMPLEMENTATION OF FORWARD-BACKWARD LMS ADAPTIVE LINE ENHANCER

Hen-Geul Yeh and Tien M. Nguyen

Jet Propulsion Laboratory
California Institute Technology
4800 Oak Grove Drive
Pasadena, CA 91109-8099

ABSTRACT

An efficient implementation of the Forward-Backward least Mean Square (FBLMS) adaptive line enhancer is presented in this paper, Without changing the characteristics of the FBLMS adaptive line enhancer [1], the proposed implementation technique reduces multiplications by 25°/0 and additions by 12.5°/0 in two successive time samples in comparison to that of the direct implementation in both prediction and weight control. The proposed FBLMS architecture and algorithm can be applied to digital receivers for enhancing Signal-to-Noise Ratio (SNR) to allow fast carrier acquisition and tracking in both stationary and nonstationary environments.

1. 1NTROD1JCTION

Adaptive line enhancers (ALE) are useful in many areas including time-domain spectral estimation for fast carrier acquisition [2-4]. For example, a fast carrier acquisition technique [2,7] as shown in Figure 1 will be very useful for deep-space mission, especially in a nonstationary environment or emergencies. Figure 1 shows the block diagram of using an ALE in the digital receiver for both acquisition and tracking. First, the receiver is in the acquisition mode, Second, when the uplink carrier is acquired as indicated by the lock detector, the switch is shifted to the

tracking position and the tracking process takes over immediately. With this acquisition scheme, the uplink carrier can be acquired by a transponder in seconds (as opposed to minutes for the Cassini transponder). Although devised to support the space mission, the architecture of FBLMS and the associated algorithm proposed in this article are also applicable to other systems, including fixed-ground and mobile communication systems. Note that this proposed ALE scheme in the receiver needs residual carrier, and does not work for suppressed carrier cases.

A conventional ALE system using least mean square (LMS) algorithm is depicted in Figure 2., The analysis of the ALE for enhancing the SNK to allow fast acquisition is given in [2]. The block diagram of a FBLMS adaptive line enhancer is shown in Figure 3. The performance analysis of the FBLMS adaptive line enhancer is provided in [1], The FBLMS adaptive line enhancer algorithm enjoys approximately half t he misadjust ment in comparison to that of the LM S algorithm [1]. However, it requires about twice the number of multiplications and additions of the LMS algorithm. In this paper, an efficient implementation of the fast FBLMS algorithm is presented. This fast algorithm provides the same speed of convergence as that of the LMS algorithm, and provides the same misadjustment as that of the FBLMS adaptive line enhancer, but requires less multiplications and additions. The computational reduction is achieved by grouping two successive predictor computations together and computing weight adaptation at every other sampling time [5]. By using a radix-2 structure to manipulate time samples, redundant computations embedded in two successive time samples can be removed via a new structure of the fast FBLMS algorithm.

This paper is organized as follows. The FBLMS algorithm is reviewed in Section 11. The fast FBLMS algorithm is derived and proposed in Section 111 The fast FBLMS algorithm implementation is given in Section IV and simulation results are presented in Section V. Finally, conclusion is given in Section VI.

Il. FORWARD-BACKWARD LMS ADAPTIVE LINE ENHANCER ALGORITHM

The structure of the forward-backward LMS adaptive line enhancer [1] is shown in Figure 3. The forward and backward prediction errors are then defined, respectively, as follows:

$$e_f(\text{t?}) = x(n) - X^T(n)W(n) \tag{1a}$$

$$e_b(n) = x(n - \text{N}) - X_b^T(n)W(n) \tag{1b}$$

where the superscript T denotes the transpose of a vector, and

$$X^T(n) = [x(n-1), x(n-2), \ldots, \text{x1/A-})])] \tag{It}$$

$$X_b^T(n) = [\text{x(H-A'-t 1), X(1?-- N + 2),.,.,X(H)}] \tag{id}$$

$$W^T(n) = [w_1(n), w_{2\,(n)}, \ldots . w_N(n)] \tag{1e}$$

In any gradient algorithm, the coefficient vector W(n) is updated using

$$W(n+1) = W(n) - \mu\, \hat{\nabla}\{e(n)^2\} \tag{2a}$$

where μ is the adaptive step size and the $\hat{\nabla}\{e(n)^2\}$ is the estimated gradient of the surface of $E\{e(n)^2\}$. Note that E{. } denotes the expected value. In the forward-backward algorithm, $e(n)^2 = e_f(n)^2 + e_b(n)^2$, and the gradient estimate is chosen as

$$\hat{\nabla}\{e(n)^2\} = -[e_f(n)X(n) + e_b(n)X_b(n)]. \tag{2b}$$

It is shown in [1] that Eq.(2b) is an unbiased estimator of the gradient. This leads to the coefficient update Eq.

$$W(n+1) = W(n) + \mu[e_f(n)X(n) + e_b(n)X_b(n)]. \tag{2c}$$

This means that W(n+1) ≅ W(n) in steady state when both forward and backward errors arc approaching zero.

111. THE FAST FORWARD-BACKWARD LMS ALGORITHM

The fast FBLMS algorithm is derived in this section by using the radix-2 algorithm on time samples. Both predictor and weights update sections are provided in detail

A, PREDICTOR SECTION

We consider the computation of two successive predictions in both forward and backward directions with fixed weight coefficient W(n- 1). After regrouping, even and odd terms, the forward predictor is obtained [5] and given in Eq. (3).

$$\begin{bmatrix} \hat{d}_f(n-1) \\ \hat{d}_f(n) \end{bmatrix} = \begin{bmatrix} X^T(n-1) \\ X^T(n) \end{bmatrix} W(n-1) = \begin{bmatrix} A^T & B^T \\ C^T & A^T \end{bmatrix}_n \begin{bmatrix} W_0 \\ W_1 \end{bmatrix}_{n-1} \tag{3a}$$

where

$$A^T = [x(n-2), x(n-4), \ldots, x(n-N+2), x(n-N)] \tag{3b}$$

$$B^T = [x(n-3), x(n-5), \ldots, x(n-N+1), x(n-N-1)] \tag{3c}$$

$$CT = [x(n-1), x(n-3), \ldots, x(n-N+3), x(n-N+1)] \tag{3d}$$

$$J\%'. = [w_0(n-1), w_2(n-1), \ldots, w_{N-2}-1)]^T, \tag{3e}$$

$$W_1 = [w_1(n-1), w_3(n-1), \ldots, w_{N-1}(n-1)]^T \tag{3f}$$

Similarly, backward predictor is obtained and given as follows:

$$\begin{bmatrix} \hat{d}_b(n-1) \\ \hat{d}_b(n) \end{bmatrix} \begin{bmatrix} X_b^T(n-1) \\ 1 \quad X_b^T(n) \end{bmatrix}^{-1} W(n-1) = \begin{bmatrix} F^T & G^T \\ G^T & H^T \end{bmatrix}_n \begin{bmatrix} W_0 \\ W_1 \end{bmatrix}. \tag{4a}$$

where

$$F^T = [x(n-N), x(n-N+2), \ldots, x(n-4), x(n-2)] \tag{4b}$$

$$G^T = [x(n-N+1), x(n-N+3), \ldots, x(n-3), x(n-1)] \tag{4c}$$

$$H^T = [x(n-N+2), x(17-N+4), \ldots, x(n-2), x(n)] \tag{4(I)}$$

Eqs. (3a) and (4a) are approximations by virtue of updating the weight vector only once every two cycles. The relationship between the two sequence sets { A,B,C } and { F,G,H } is given as follows:

$$F = A_r \tag{5}$$

$$\mathbf{G = c,} \tag{6}$$

$$z^{-1}H = A_r \tag{7}$$

where subscript r means the reversed order of the sequence and the z^{-1} means one delay unit of

the corresponding sequence and is equivalent to two time sample delays. Furthermore, we observe the following relationships between G, B, and C.

$$z^{-1}G = B_r \tag{8}$$

$$z^{-1}C = B \tag{9}$$

After performing the appropriate computation, Eq. (4a) can be rewritten as follows:

$$\begin{bmatrix} \hat{d}_b(n-1) \\ \hat{d}_b(n) \end{bmatrix} = \begin{bmatrix} G^T(W_0 + W_1) + (F - G)^T W_0 \\ G'(J'V_0 + W_1) - (G - H)^T W_1 \end{bmatrix} \tag{10}$$

The computation of Eq. (4a) requires 2 inner products of length N, while that of Eq. (10) requires only 3 inner products of length N/2 and N/2 additions to perform $W_0 + W_1$. Similarly, by combining Eqs. (5-9), Eq. (3a) can be rewritten as follows

$$\begin{bmatrix} \hat{d}_f(n-1) \\ \hat{d}_f(n) \end{bmatrix} = \begin{bmatrix} A^T(W_0 + W_1) + (B - A)^T W_1 \\ A^T(W_0 + W_1) - (A - C)^T W_0 \end{bmatrix}$$

$$= \begin{bmatrix} A^T(W_0 + W_1) + z^{-1}(G - H)_r^T W_1 \\ A^T(W_0 + W_1) - (F - G)_r W_0 \end{bmatrix} \tag{11}$$

Clearly, the sequences (G-H) and (F-G) of Eq. (1 O) are reused again in Eq. (11) but in reversed order. The computation of Eq. (11) requires only 3 inner products of length N/2. The total number of multiplications and additions required in both forward and backward predictor sections for two successive computations is about 3N and 3 .5N, respectively. The total number of multiplications and additions required in Eqs. (1a) and (1 b) for two successive prediction section

is *4N* and 4(N- 1). Consequently, there arc about 25°/0 and 12.5°/0 savings in multiplications and additions, respectively.

B. WEIGHT UPDATE SECTION.

We consider the weight coefficient updates now. Since weights arc explicitly computed at every other time update using the look-ahead approach [6], the weight update of Eq. (2c) can be rewritten as follows:

$$W(n+1) = W(n-1) + \mu\left[e_f(n-1)X(n-1) + e_b(n-1)X_b(n-1)\right] + \mu\left[e_f(n)X(n) + e_b(n)X_b(n)\right]$$

$$= W(n-1) + \begin{bmatrix} X(n) & X(n-1) \end{bmatrix}\begin{bmatrix} \mu e_f(n) \\ \mu e_f(n-1) \end{bmatrix} + \begin{bmatrix} X_b(n) & X_b(n-1) \end{bmatrix}\begin{bmatrix} \mu e_b(n) \\ \mu e_b(n-1) \end{bmatrix} \qquad (12)$$

By combining Eqs.. (5-9), Eq. (12) is rewritten as follows:

$$\begin{bmatrix} W_0 \\ W_1 \end{bmatrix}_{n+1} = \begin{bmatrix} W_0 \\ W_1 \end{bmatrix}_{n-1} + \begin{bmatrix} C & A \\ A & B \end{bmatrix}\begin{bmatrix} \mu e_f(n) \\ \mu e_f(n-1) \end{bmatrix} + \begin{bmatrix} G & F \\ H & G \end{bmatrix}\begin{bmatrix} \mu e_b(n) \\ \mu e_b(n-1) \end{bmatrix}$$

$$= \begin{bmatrix} W_0 \\ W_1 \end{bmatrix}_{n-1} + \mu\begin{bmatrix} A(e_f(n) + e_f(n-1)) - (F-G)_r e_f(n) \\ A(e_f(n) + e_f(n-1)) + z^{-1}(G-H)_r e_f(n-1) \end{bmatrix}$$

$$+ \mu\begin{bmatrix} G(e_b(n) + e_b(n-1)) + (F-G)e_b(n-1) \\ G(e_b(n) + e_b(n-1)) - (G-H)e_b(n) \end{bmatrix} \qquad (13)$$

The vectors (F-G) and (G-H) are once more employed in Eq. (13). Notice that the term $\mu[A(e_f(n) + e_f(n-1)) + G(e_b(n) + e_b(n-1))]$ is computed only once and the sum is applied to both W. and W_1 for updates. The total number of multiplications and additions in Eq. (13) is about 3N and 3.5N respectively. However, the total number of multiplications and additions of Eq. (2c) for

two adaptations is 4N and 4(N- I). Consequently, 25°/0 of multiplications and12.5% of additions arc saved by using Eq. (13) in comparison with that of Eq. (2c).

IV. IMPLEMENTATION

The architecture of the fast FBLMS algorithm is depicted in Figure 4. A switching circuit is employed after the adaptive line enhancer and the switch rate (from C to A or from A to C) is the same as the sampling rate. 'l'he switching circuit is switched between points C and A alternately. Sequences C and A arc generated at a rate of 1/(2T) accordingly. The sequence B is a delayed version of the sequence C. By using a radix-2 structure, sequences {B-A} and {A-C} are then generated at the upper- and lower-lag, respectively. By using the sequence {B-A}, inner products $(B\text{-}A)^T W_1$ and $z^{-1}(G\text{-}H)^T W$, are generated at the upper- and lower-lag of the upper forward-backward tapped-delay-line structure, respectively. Similarly, by using the sequence {A-C}, inner products $(A\text{-}C)^T W_0$ and $(F\text{-}G)^T W_0$ are generated at the upper- and lower-lag of the lower forward-backward tapped-delay-line structure, respectively. Note that vectors F, G, and H are defined in Eqs. (5), (6), and (7), respectively. Inner products of $A^T(W_0+W_1)$ and $G^T(W_0+W_1)$ are computed at the top- and bottom-portion of the fast FBLMS architecture, respectively. Finally, forward errors {$e_f(n)$ and tin-1)} and backward errors { $e_b(n-1)$ and $e_b(n-2)$ } are computed at the right hand side of Figure 4. In order to subtract the term of $z^{-1}(G\text{-}H)^T W_1$ and form the backward error, a delay unit is applied to the output branch of the inner product of $G^T(W_0+W_1)$. Consequently, the corresponding backward error is delayed from $e_b(n)$ to $e_b(n-2)$. Notice that this radix-2 structure concept can be applied again to the upper and lower forward-

backward taped-delay-line portion of the fast FBLMS algorithm to further reduce the number of multiplications and additions.

Although the fast FBLMS architecture shown in Figure 4 is more complex than the FBLMS shown in Figure 3, the structure is still very regular. In fact, the fast FBLMS architecture is consisted of radix-2, forward LMS, and FBLMS structures. The increased complexity over the FBLMS algorithm is not significant, therefore the fast FBLMS algorithm then can be easily implemented with digital signal processors.

V. SIMULATION RESULTS

An adaptive line enhancer with 6-weight (N= 6) is chosen as an example. The input signal is a sinusoid of frequency f_o contaminated by white noise. Computer simulations are conducted for the misadjustment calculation by using forward LMS, FBLMS, and fast FBLMS algorithms. The misadjustment [1] is computed after convergence as follows

$$M = \frac{\text{Extra Output Power Due to Weight Jittering}}{\text{Minimum Output Power}} .$$

$$= \frac{E[\Delta(n)^T \phi(x,x) A(n)]}{E[e(n)^2]_{opt}} \tag{14}$$

where $\Delta(n)=W(n)-W_{opt}$ (15)

$$\phi(x,x)=E[X(n)X^T(n)] \tag{16}$$

$$E[e(n)^2]_{opt} = E[x(n)^2] - W_{opt}^T E[x(n)X(n)] \tag{17}$$

Table 1 shows the measured misadjustments for various values of Signal-to-Noise power Ratio (SNR) at step size $\mu = 2^{-8}$. Apparently, the excess error power for both FBLMS and fast FBLMS algorithms is approximately half that of the forward Lh4S algorithm at the 10 dB SNR. The improvement of the misadjustment by using both FBLMS and fast FBLMS algorithms over that of the forward LMS algorithm is limited at the SNR around O dB. However, the misadjustment of the fast FBLMS algorithm is about the same as that of the FBLMS algorithm. Furthermore, it is observed in "1'able 1 that at higher SNR the misadjustment increases (for a given step size $\mu = 2$-8).

This is because the minimum output error power decreases much mor c rapidly than the extra output power due to weight jittering, as depicted by Eq. (14). This high misadjustment is significantly reduced when the step size μ is cut to 2^{-10} as shown in "l-able 11.

Table 11 shows the measured misadjustments foi various values of the step size and the frequency f_0 at SNR = 10 dB. Apparently, the excess error power foi both FBLMS and fast FBLMS algorithms is approximately half that of the forward LMS algorithm at the step size $\mu = 2^{-8}$ and $\mu = 2^{-10}$. The misadjustment is much r-educed when the step size is small by using any algorithm. Again, the misadjustment of the fast FBLMS algorithm is about the same as that of the FBLMS algorithm. The $E[e(n)^2]_{opt}$ used to derive the misadjustrnent is computed by using 500 samples in each run. The misadjustment results listed in Tables 1 and II were obtained by averaging 100 runs of the excess error power curves after convergence has been achieved.

Table 1 A Comparison Between the Misadjustment Powers of Three Algorithms at $\mu = 2^{-X}$

SIR	f_0	% Misadjustment		
		Forward LMS	FBLMS	Fast FBLMS
0	.1	3.04	2.75	2,75
3	.1	3.74	2.84	2,93
10	.1	32.50	13.77	16.95

'l'able 11. A Comparison Between the Misadjustment Powers of Three Algorithms Using fixed SNR = 10 dB with different μ.

μ	f_0	% Misadjustment		
		Forward LMS	FBLMS	Fast FBLMS
2^{-8}	.1667	31.34	14,47	16.03
2^{-8}	.1	32.5	13.77	16.95
2^{-10}	.1667	3.06	2.05	1.99
2^{-10}	.1	2.33	1.24	1.30

Figures 5(a), (b), and (c) show a typical excess error power versus n plot at $f_0 = 1/6$, step size $= 2^{-8}$, and SNR = 10 dB for the forward LMS, FBLMS, and fast FBLMS algorithms, respectively. Figure 5(d) shows the excess error power at the steady state. It is clear that the performance of the fast FBLMS algorithm is about the same as that of the FBLMS algorithm.

VI. CONC1,1JS1ON

The fast forward-backward LMS algorithm presented in this paper shows that the number of arithmetic operations in [1] can be reduced without degrading its performance. In the forward - backward predictor section, 25% of multiplications and 12.5% of additions are saved in each two successive operations. Similarly, in the weight control section, 25°/0 of multiplications and 12.5°/0 of additions are saved in each two adaptations. Simulation results indicate that misadjustment improvements for both FBLMS and fast FBLMS algorithms over the conventional LMS algorithm is about 50°/0 at high SNR. When the SNR is low, the misadjustment improvement for both FBLMS and fast FBLMS algorithms over the conventional LMS algorithm is less than 50%.

Notice that this fast forward-backward LMS algorithm is well suited for implementation on ASICs and digital signal processors. This implementation method can be generalized by using higher than 2 steps of look-ahead, Further computational savings are possible with limited cost on controlling appropriate data flow. This fast FBLMS adaptive line enhancer can be easily integrated together with either a conventional Voltage Controlled Oscillator (VCO) in a closed loop for acquisition/tracking as present deep space transponder, or Numerical Control Oscillator (NCO) in an open loop scheme for acquiring and tracking the carrier signal as future deep space transponders.

References

[1] Y.C. Lim and C.C. Ko, "Forward-backward LMS adaptive line enhancer," IEEE Trans Circuits Syst., Vol. 37, no.7, pp.936-940, July, 1990.

[2] H. G. Yeh and T. M. Nguyen, 'Adaptive line enhancers for fast acquisition," TDA Progress Report 42-119, pp. 140-159, Nov., 1994

[3] L. Griffiths, "Rapid measurement of digital instantaneous frequency," 1111311 Trans. Acoust., Speech, Signal Processing, Vol. ASSP-23, No. 2, pp. 207-222, April 1975.

[4] J. T. Richard et al., "A performance analysis of adaptive line enhancer - augmented spectral detectors," IEEE Trans. ASS]), vol. ASSP-29, pp,694-701, June 1981.

[5] Z. J. Mou and P. Duhamel, "Short-length FIR filters and their use in fast nonrecursive fltering," IEEE Trans. Acoust., Speech, Signal Process., vol. AS SP-39, pp. 1322-1332, June 1991.

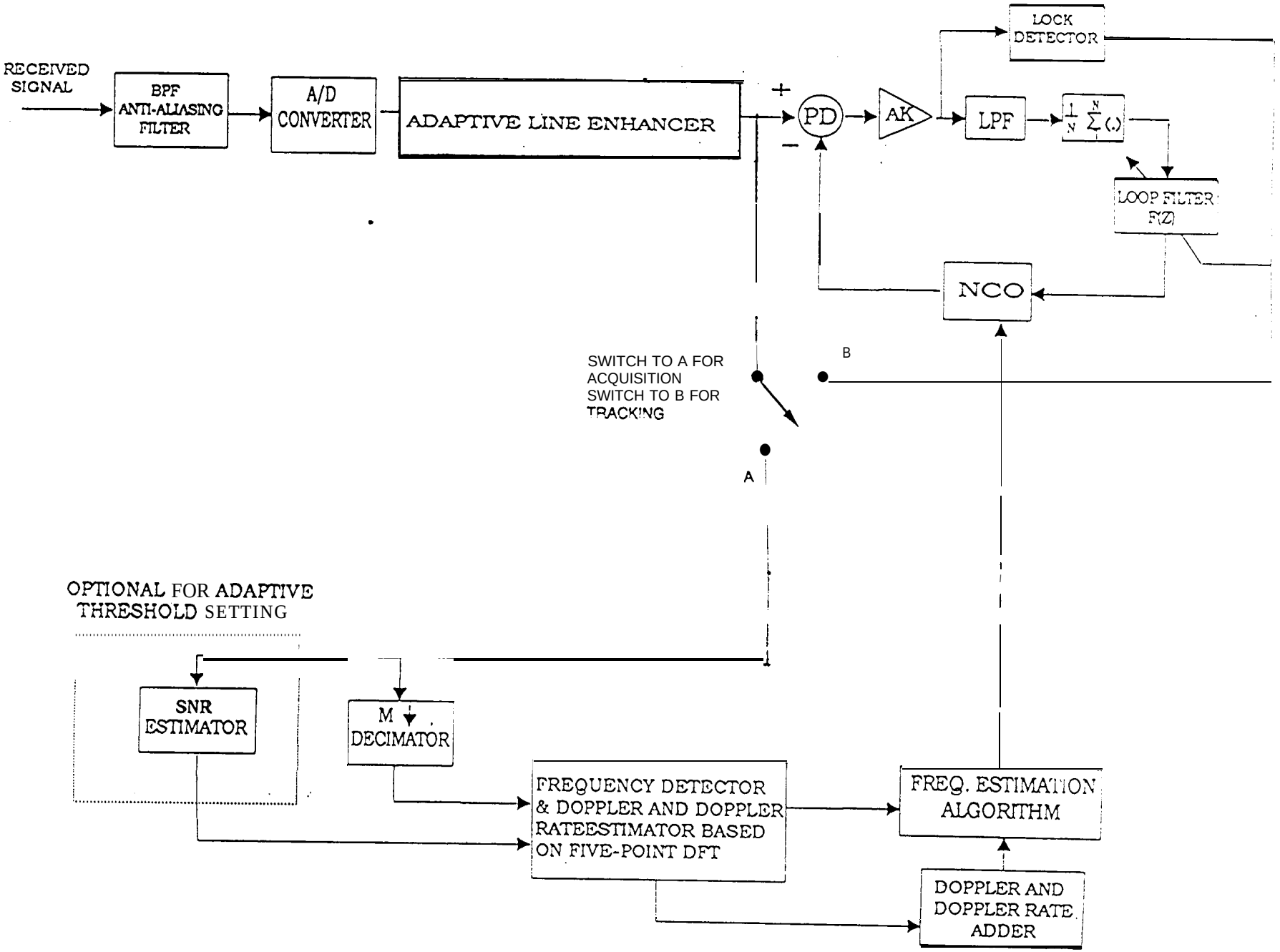
[6] K. K. Parhi and D. G. Messerschmitt, "Pipeline interleaving and parallelism in recursive digital filters - Part 1: pipelining using scattered look-ahead and decomposition," IEEE Trans. Acoust., Speech, Signal Process., vol. ASSP-37, pp. 1099-1117, July 1989.

[7] Tien M. Nguyen, H. G. Yeh, and Loc V. Lam, "A new carrier frequency acquisition technique for future digital transponders," to be published

ACKNOWLEDGMENT

The authors wish to acknowledge the valuable support of Arthur W. Kermode, Lance A. Riley, and Sami M. Hinedi. Special thanks to Victor A. Vilnrotter for his review, collection, and discussions.

Figure 1. The block diagram of using ALE in the digital receiver for both acquisition and tracking.



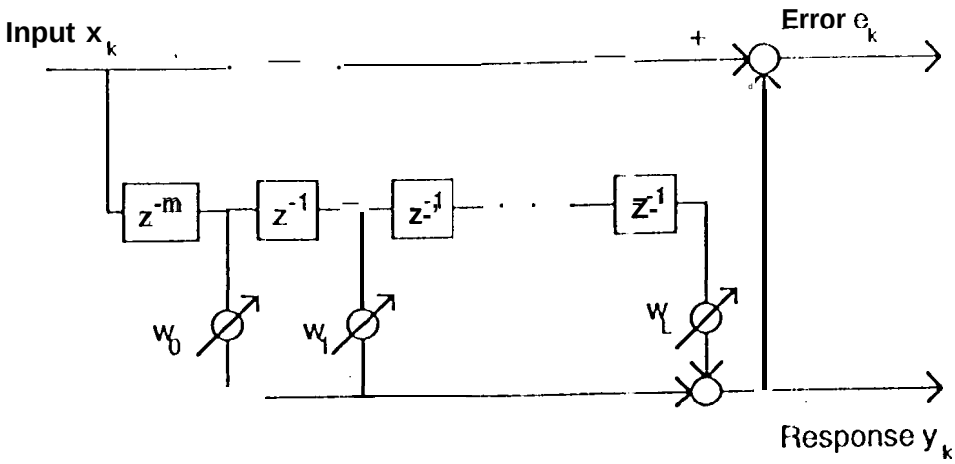


Figure 2. The architecture of conventional ALE.

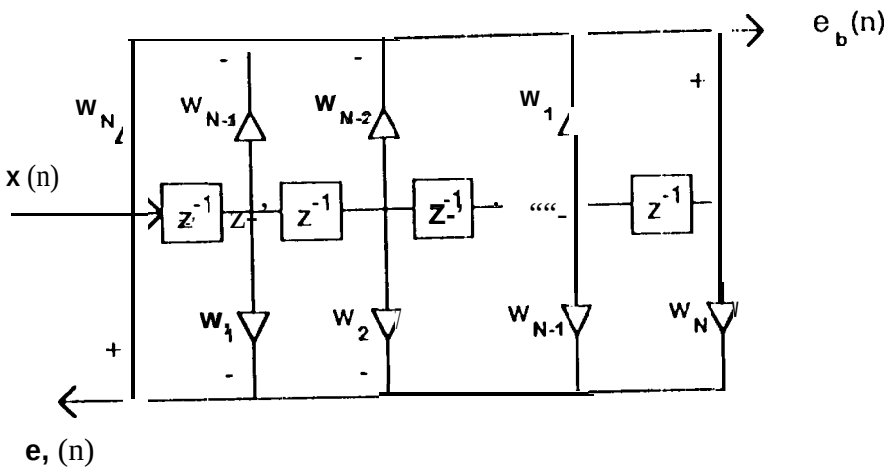


Figure 3. The structure of the FBLMS adaptive line enhancer.

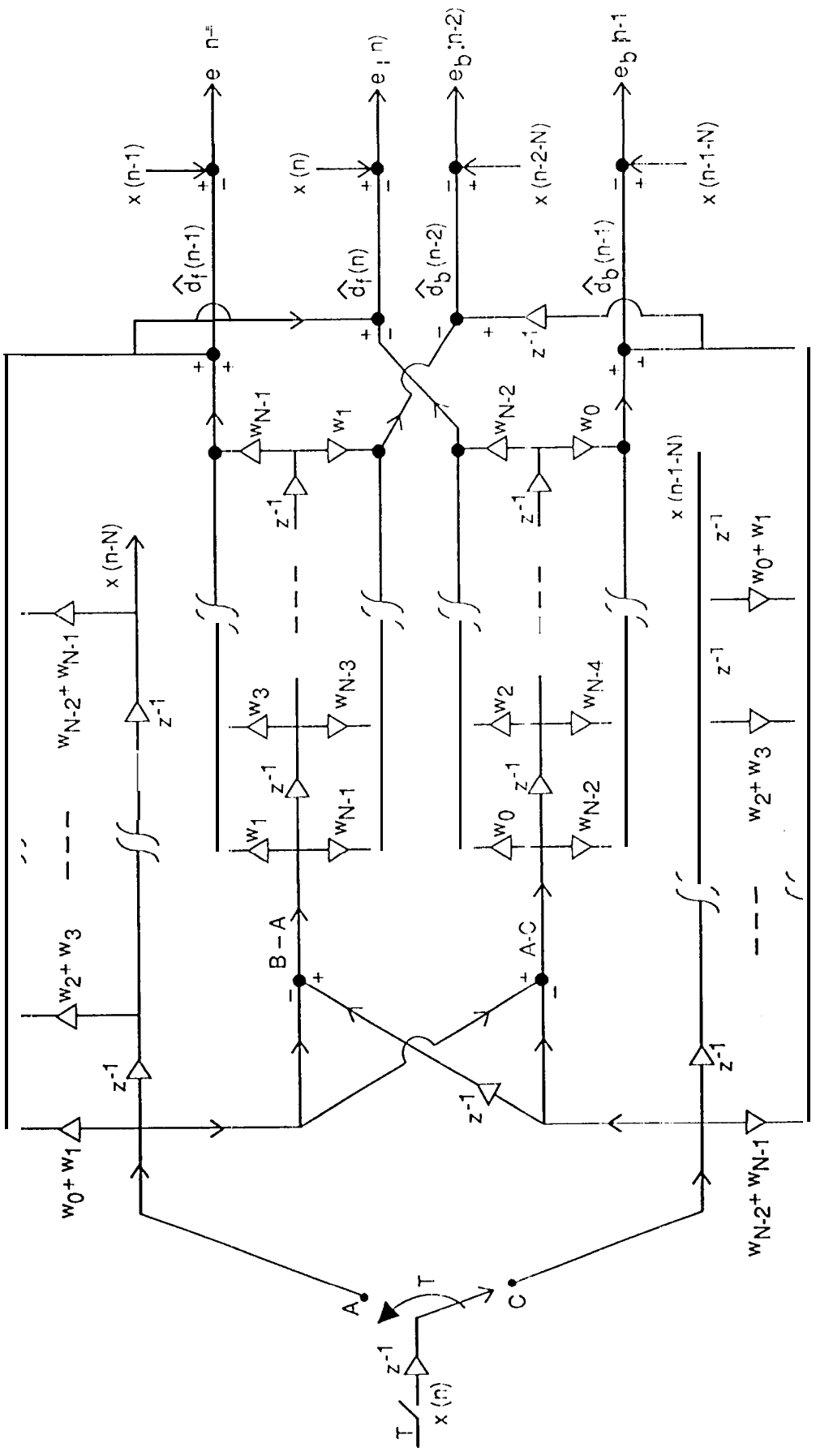
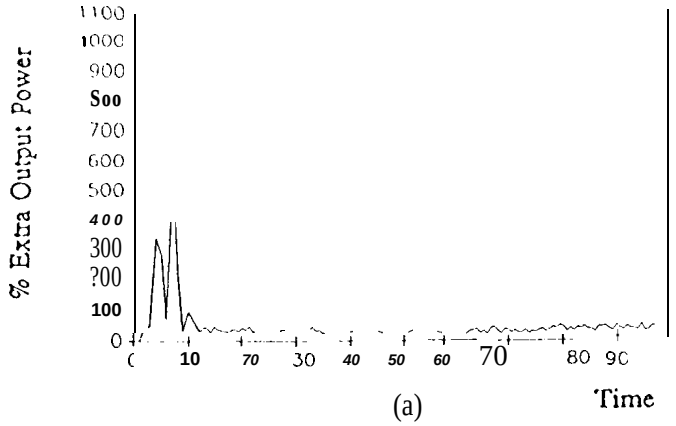
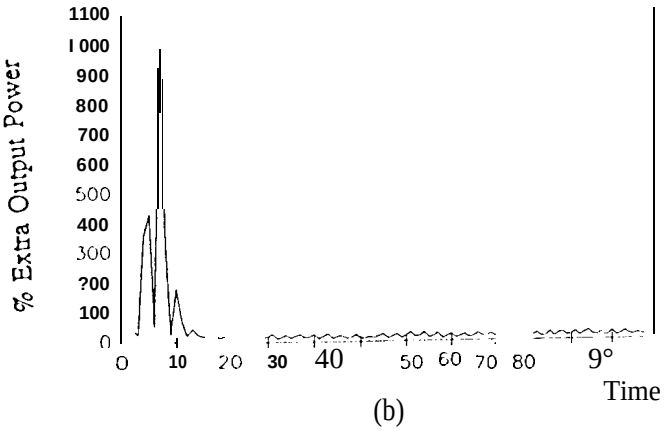


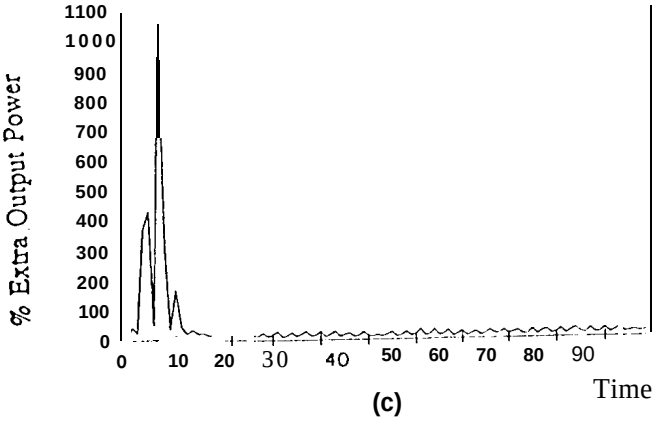
Figure 4. The architecture of the fast FBLMS algorithm.



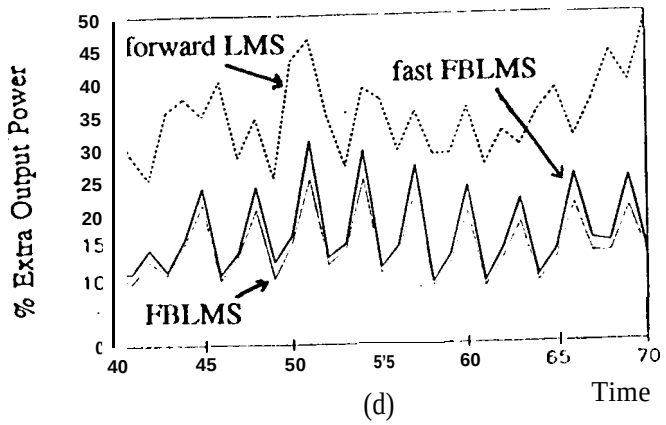
(a)



(b)



(c)



(d)

Figure 5. A typical excess error power versus n plot by using the (a) forward I.MS, (b) FBLMS, (c) fast FBLMS algorithm; (d) the steady state comparison